

**INTERFACE CIRCUIT FOR CARD-TYPE MEMORY,  
ASIC INCLUDING INTERFACE CIRCUIT, AND  
IMAGE FORMING APPARATUS INCLUDING ASIC**

**5 CROSS-REFERENCE TO RELATED APPLICATIONS**

The present document incorporates by reference the entire contents of Japanese priority documents, 2002-382155 filed in Japan on December 27, 2002 and 2003-414817 filed in Japan on December 12, 2003.

10

**BACKGROUND OF THE INVENTION**

**1) Field of the Invention**

The present invention relates to an interface circuit for a card-type memory such as a secure digital (SD) card that is detachable and requires access by sectors, and to an application specific integrated circuit (ASIC) including the interface circuit, and an image forming apparatus including the ASIC.

**2) Description of the Related Art**

One approach to update software for equipment such as printers, copying machines, and multifunction peripherals (MFP) is to load the software from a memory card or download the software through a host interface of a network. However, in many cases, e.g., in field support, the host interface can not be used.

To allow the use of the host interface, some equipment are

provides with a memory card interface. The memory card is handled in the same manner as that of random access memory (RAM). In other words, when the memory card is used, the interface does not require to be initialized. Besides, a program can be executed on the memory  
5 card. However, the memory cards have a problem in that they have low capacity of at most 4 megabytes, and they are not readily available in the market these days.

Japanese Patent Application Laid Open No. H11-242596 (see Fig. 1) discloses SD cards that can be used instead of the memory  
10 cards. The SD cards are attracting attention as portable media like floppy disks (FD). The SD card has a larger capacity per unit size, so that they useful for recording and reproducing image data or audio data. If the access to the SD card is restricted to a read-access, a basic  
15 input/output system (BIOS) is not required. In addition, only required data can be read out according to the access from a central processing unit (CPU), and therefore, a program can be executed on the SD card without data copying to a random access memory (RAM). If a communication error occurs, a communication speed is automatically  
20 reduced step by step to a speed at which no error occurs, then the processing is continued. Thus, it is possible to perform data communications without changing software and hardware.

However, in an SD card interface with a buffer size only by one sector, if a program extends over a plurality of sectors due to a jump instruction or the like, an overhead required for reading data from the  
25 card-type memory is largely affected, which causes an execution speed

of the program to be low.

#### SUMMARY OF THE INVENTION

It is an object of the present invention to solve at least the  
5 problems in the conventional technology.

An interface circuit capable of allowing transmission of data from a detachable card-type memory, which requires access by sectors, to an electronic device, according to one aspect of the present includes a reading unit that reads data for a plurality of sectors from the  
10 card-type memory; a buffer that stores the data read and has a capacity to store data for a plurality of sectors; a receiver that receives from the electronic device a read-access for data stored in the buffer; a data checker that decides whether data corresponding to the read-access exists among the data stored in the buffer; and a transmitter that  
15 transmits the data from the buffer to the electronic device when the data checker decides that data corresponding to the read-access exists among the data stored in the buffer.

An application specific integrated circuit and an image forming apparatus according to other aspects of the present invention employ  
20 the interface circuit according to the present invention.

The other objects, features, and advantages of the present invention are specifically set forth in or will become apparent from the following detailed descriptions of the invention when read in conjunction with the accompanying drawings.

## BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram of an image forming apparatus according to an embodiment of the present invention;

Fig. 2 is a detailed block diagram of an SD card interface used  
5 in the image forming apparatus;

Fig. 3 is a flowchart of procedures for initialization of the image forming apparatus;

Fig. 4 is a flowchart of procedures for initialization of the SD card;

10 Fig. 5 is a flowchart of procedures for acquiring a card size;

Fig. 6 is an example of a memory map of the SD card; and

Fig. 7 is a flowchart of processing for a read-access from a  
CPU.

## 15 DETAILED DESCRIPTION

Exemplary embodiments of an interface circuit for a card-type memory, an application specific integrated circuit, and an image forming apparatus according to the present invention are explained in detail below with reference to the accompanying drawings.

20 Fig. 1 is a block diagram of an image forming apparatus according to an embodiment of the present invention. This image forming apparatus includes an ASIC 10 that includes a plurality of application functions such as an image input/output function, an image processing function, and data communications. The application  
functions are designed to share a memory 6 and a hard disk in a hard  
25

disk drive (HDD) 1 as common resources. The ASIC 10 is connected with the HDD 1, a physical layer (PHY) device (which is, for example, a network device) 2, a PHY device (which is, for example, a USB device) 3, and an SD card 4 (hereinafter, "card 4"). The ASIC 10 is also 5 connected with a CPU 5, the memory 6, a printer engine 20, and an Institute of Electrical and Electronics Engineers (IEEE) 1284 device (not shown) which are provided in the body of the image forming apparatus.

The HDD 1 stores image data and programs, the CPU 5 controls the operation of the image forming apparatus, and the memory 6 is 10 randomly accessible. The printer engine 20 functions as an imaging unit that forms a visible image on a recording material, and can form an image based on image data transferred from an external device through an IEEE 1284 interface 17, image data transferred from the PHY devices 2 and 3, and image data stored in the HDD 1 and the card 4. 15 By forming a system with the printer engine 20, an image forming apparatus such as a printer, a copying machine, and a facsimile each including the ASIC 10 is realized.

More specifically, the ASIC 10 includes a memory arbiter 30, an HDD interface 11 and a direct memory access (DMA) controller 21 for 20 connecting between the memory arbiter 30 and the HDD 1, and a network interface 12 and a DMA controller 22 for connecting between the memory arbiter 30 and the network device 2. The ASIC 10 also includes a USB interface 13 and a DMA controller 23 for connecting 25 between the memory arbiter 30 and the USB device 3, and an SD card interface 14 and a DMA controller 24 for connecting between the

memory arbiter 30 (and a memory controller 16) and the card 4. The ASIC 10 further includes a CPU interface 15 for connecting between the memory arbiter 30 and the CPU 5, the memory controller 16 for connecting between the memory arbiter 30 (and SD card interface 14) 5 and the memory 6, the IEEE 1284 interface 17 for connecting between the memory arbiter 30 and the IEEE 1284 device (not shown), and a printer engine interface 18 for connecting between the memory arbiter 30 and the printer engine 20.

In addition to the applications for the hard disk, the network, or 10 the like, the SD card interface 14 is connected with a data transfer path by the DMA controller 24 through the memory arbiter 30 and with a path to the memory controller 16 for random access. In such a configuration, data transfer can be performed not only from the card 4 to the memory 6 but also between the card 4 and HDD 1 through the 15 HDD interface 11, the network device 2 through the network interface 12, or the USB device 3 through the USB interface 13.

Fig. 2 is a detailed block diagram of the SD card interface 14. This SD card interface 14 includes an SD card controller 141 that performs actual data communications with the card 4 according to the 20 specification of the card 4. A control circuit 145 is a circuit block that receives a control command issued from the CPU 5 to the SD card controller 141 or transfers commands from the SD card controller 141 to the CPU 5, and switches to a multiplexer 144 that selects between a DMA interface 142 and a RAM access interface 143 for data transfer to 25 the card 4.

The DMA interface 142 is a circuit block that connects between the SD card controller 141 (and the multiplexer 144) and the DMA controller 24 of Fig. 1, and that provides matching between the DMA interface 142 and the SD card interface 14 to allow data transfer by 5 DMA. The RAM access interface 143 is a circuit block that connects between the memory (RAM) controller 16 or the CPU interface 15 and the SD card controller 141 (and the multiplexer 144), and that provides matching between the RAM access interface 143 and the SD card interface 14.

10 The RAM access interface 143 includes a buffer (RAM) 143a having a capacity to hold data of a plurality of sectors (512 bytes×N sectors). Data in the buffer 143a is transferred to the card 4. If a read-access is received from the CPU 5 relating to data stored in the buffer 143a, the data is read from the buffer 143a and not from the card 15 4. Consequently, the access to the card 4 is suppressed to a minimum.

The processing of initialization in the image forming apparatus according to the embodiment is explained below. The card 4 stores a boot program. Generally, the system program such as the boot 20 program is installed in the memory 6 such as read only memory (ROM) and the memory 6 is placed on a board such as a controller board. However, in the present embodiment, the boot program is stored in the card 4 in advance to suppress a unit cost of pits, and this allows reduction of developing costs. Moreover, the system program stored in 25 the card 4 can be updated or edited more easily than those in the ROM.

Therefore, upgrading of the image forming apparatus and a change in the program by a correction program can easily be realized.

Fig. 3 is a flowchart of the procedures for initialization of the image forming apparatus. The initialization is performed when power of the image forming apparatus is turned on. In the image forming apparatus, the CPU 5 releases the system reset (step S301). After the startup of the CPU 5, the boot program stored in the card 4 is read and executed (step S302). More specifically, the boot program stored in the card 4 is copied to the RAM. Then, the boot program copied to the RAM is executed (step S303).

The boot program is generally stored in the card 4, but all the system program including the boot program may be stored in a rewritable nonvolatile memory such as a flash memory on a controller board, and a new-version system program is stored in the card 4 to update the system program in the nonvolatile memory by the system program stored in the card 4.

Fig. 4 is a flowchart of the procedures for initialization of the card 4 by the SD card interface 14. As the initialization is sequentially operated, it is not so difficult to operate unless the procedures are executed in wrong order. However, the problem is occurrence of an error due to cases such that a card is not inserted into the image forming apparatus and a card that is not allowed to use is inserted thereto. If the card is not inserted, its insertion can be continuously confirmed during a predetermined period.

Occurrence of an error may be suppressed by restricting the

purpose of using the initialization for the hardware in the SD card interface 14. In other words, the purpose is restricted to some specific works by a service engineer such as update of software and execution of a self-diagnostic program.

5       The procedures for the initialization of the SD card by the SD card interface are explained with reference to Fig. 4. At first, it is determined whether the card 4 has been initialized (step S401). If the card 4 has not been initialized, reset of SD card interface is released (step S402). It is checked whether the card 4 has been inserted (steps 10 S403, S404), and if the card 4 has been inserted and it is recognized, the process proceeds to step S405. On the other hand, if the card 4 can not be recognized, the process returns to step S401.

In the processing at step S405 and thereafter, an SD clock is set (step S405), the card 4 is initialized (step S406), an operating voltage is set (step S407), a card identification number (ID) is acquired (step S408), a card address is acquired (step S409), a card size is acquired (step S410), a block length is set (step S411), a data bus width is set (step S412), a card status is acquired (step S413), an initialization-end flag indicating that the card has been initialized is set (step S415). If 20 an error occurs, a card not-yet-inserted flag indicating that the card has not yet been inserted is set (step S414), and the process proceeds to step S415.

Fig. 5 is a flowchart of procedures for acquiring the card size. Fig. 6 is an example of a memory map of the SD card. The memory 25 map includes a partition table, a boot sector, a file allocation table (FAT),

a root directory, and a user data area. The processing for acquiring the card size at step S410 requires an offset address of a user area according to a storage capacity of the SD card.

In the processing for acquiring the card size, at first, a user area  
5 size (C\_SIZE) of the SD card is acquired (step S501). A multiplying factor of a device size of the SD card (C\_SIZE\_MULT) is acquired (step S502). A card size (card\_size) is calculated by the following equation (step S503):

$$\text{card size} = \text{C\_SIZE} / [2^2(9 - \text{C\_SIZE\_MULT})].$$

10 An offset address (sd\_offset) according to the card size (card\_size) is determined (step S504).

The area size of the user data is obtained by performing acquisition of the card size in the above manner. A nominal size of the card 4 is obtained from the user area size to obtain the offset address  
15 (sd\_offset). Only the user area is open to the CPU 5 in areas of the SD card. Therefore, an address as a target to be read out is obtained by adding the offset address (sd\_offset) to an address (segment address: rd-adr) accessed by the CPU 5. For the access to the card 4, a sector number (sector\_num) to be read out is obtained by the  
20 following equation assuming one sector holds, for example, 512 bytes:

$$\text{sector\_num} = (\text{rd\_adr} + \text{sd\_offset}) \gg 9.$$

Fig. 7 is a flowchart of processing when a read-access to the card 4 is made from the CPU 5. This flowchart is based on a case where the card 4 has the buffer 143a holding two sectors. This circuit  
25 is also performed by sequential processing. This circuit is not started

before the initialization of Fig. 3 is ended. If the card 4 is not recognized when the initialization is ended, then the sequencer is stopped. This operation is one of the restrictions of the purposes as explained above, and therefore, it does not cause any trouble. As 5 shown in Fig. 7, only the read-access to the SD card is described. Therefore, if an access request from the CPU 5 is any request other than the read-access, an access error is returned to the CPU 5.

If a data amount of the read request from the CPU 5 is within a sector on the card 4 that corresponds to either one of the two sectors 10 held in the buffer 143a, it is determined that the buffer 143a already has data. In this case, the requested data is returned immediately from the buffer 143a to the CPU 5. At this time, the accessed sector area is defined as "area=0", and the sectors are managed so that the sector as area=0 is always the last accessed sector.

15 If the requested data is out of the range of the sectors stored in the buffer 143a, the sequencer abandons the sector data in sector area (area)=1, and transfers the sector data in area=0 to area=1 (transfer may be performed by bank switching). Corresponding sector data is read out from the card 4, and the read-out sector data is stored in 20 area=0. The data for the address requested from the CPU 5 is returned thereto. Through the operation, the CPU 5 can read out data for an arbitrary address in the card 4 without recognizing the card 4.

Fig. 7 is the flowchart of procedures for a read-access from the CPU by the SD card interface. At first, it is determined whether the 25 initialization of Fig. 1 has been ended (step S701). If the initialization

has been ended, it is determined whether the card 4 is recognized (step S702). If it is determined that no card 4 is recognized, the processing is ended. On the other hand, if the card 4 is recognized, it is determined whether an access request to the card 4 has been received 5 (step S703). If the access request has not been received, the process returns to step S703. On the other hand, if the access request has been received, it is determined whether the access request is a read request (step S704).

If the access request is the read request to the card 4, the 10 process proceeds to step S705, while if it is not the read request, the error response is executed (step S712), and the process returns to step S703.

At step S705, it is determined whether the read-access data is in the sector area=0. If it is not in the sector area=0, the process 15 proceeds to step S706, while if it is in the sector area=0, the process proceeds to step S711 where the read-access data is read from the sector area=0 and the read data is sent to the CPU 5.

At step S706, it is determined whether the read-access data is in the sector area=1. If it is not in the sector area=1, the process 20 proceeds to step S707, while if it is in the sector area=1, the data in the sector area=0 is replaced with the data in the sector area=1 (step S713), and the process proceeds to step S711 where the read-access data is read from the sector area=0 and the read data is sent to the CPU 5.

At step S707, the data in the sector area=0 is transferred to the 25 sector area=1, and a sector read command is issued to the card 4 (step

S708). The process waits until the sector read is ended (step S709). At step S709, the sector read is ended, and the sector data is copied to the sector area=0 (step S710). Then, the read-access data in the sector area=0 is read, the read data is sent to the CPU 5 (step S711),  
5 and the process returned to step S703.

Thus, even if the access from the CPU 5 is made across the sectors of the card 4, the data is maintained for a predetermined period, which allows reduction of overhead for data read in execution of a program over the boundary between sectors and in execution of a small  
10 size sub-function. Accordingly, the execution speed of the programs can be improved.

It has been mentioned above that the buffer includes an area for two sectors, but the buffer may include an area for three or more sectors.  
15

According to the present invention, it is possible to improve the execution speed of the programs.

Although the invention has been described with respect to a specific embodiment for a complete and clear disclosure, the appended claims are not to be thus limited but are to be construed as embodying  
20 all modifications and alternative constructions that may occur to one skilled in the art which fairly fall within the basic teaching herein set forth.